Assignment 1 Programming Fundamentals

Ahmed Waleed

College of Interdisciplinary Studies, Zayed University

ICS220-22527: Program. Fund.

Professor: Sujith Mathew

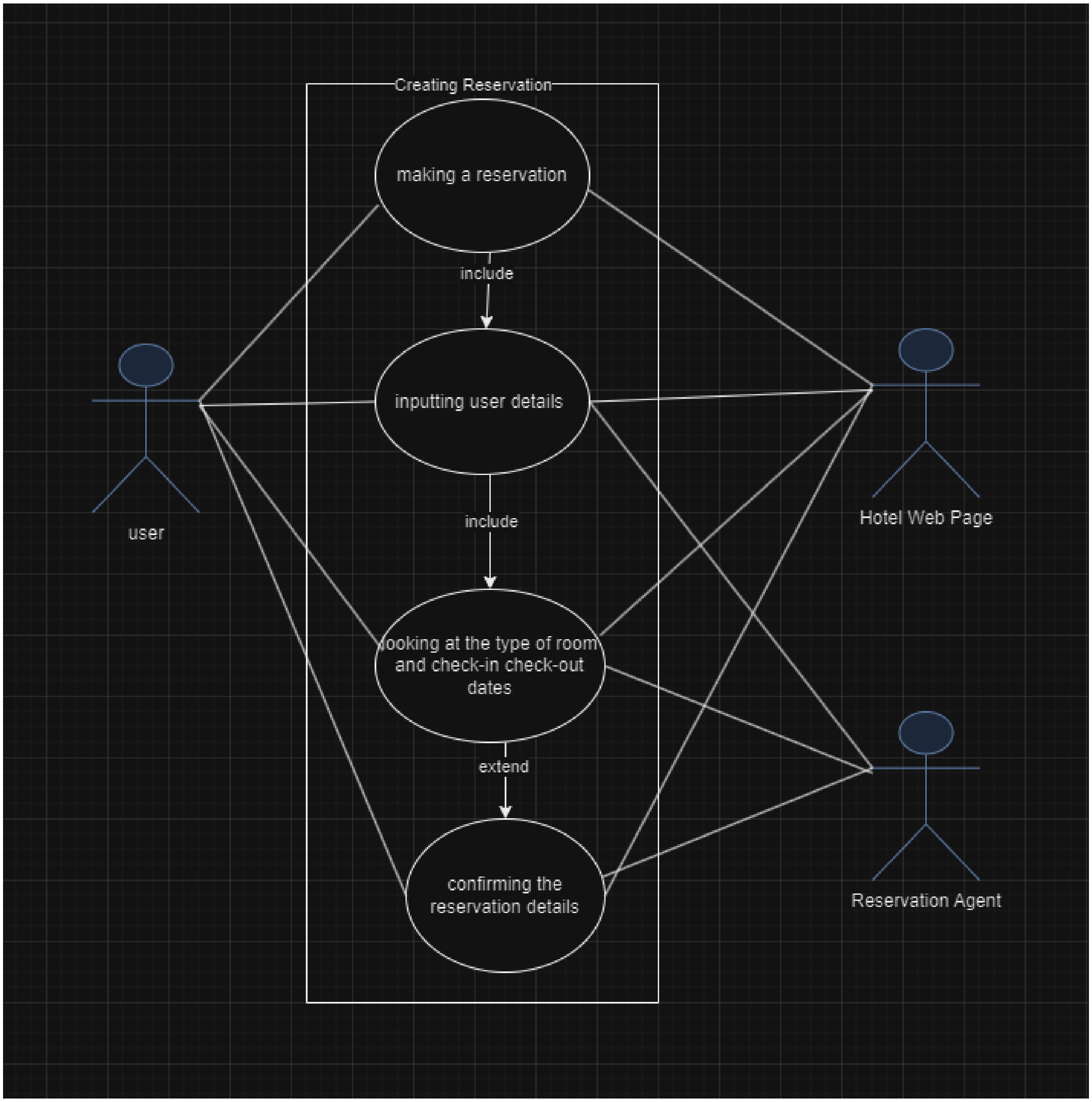
September 30, 2024

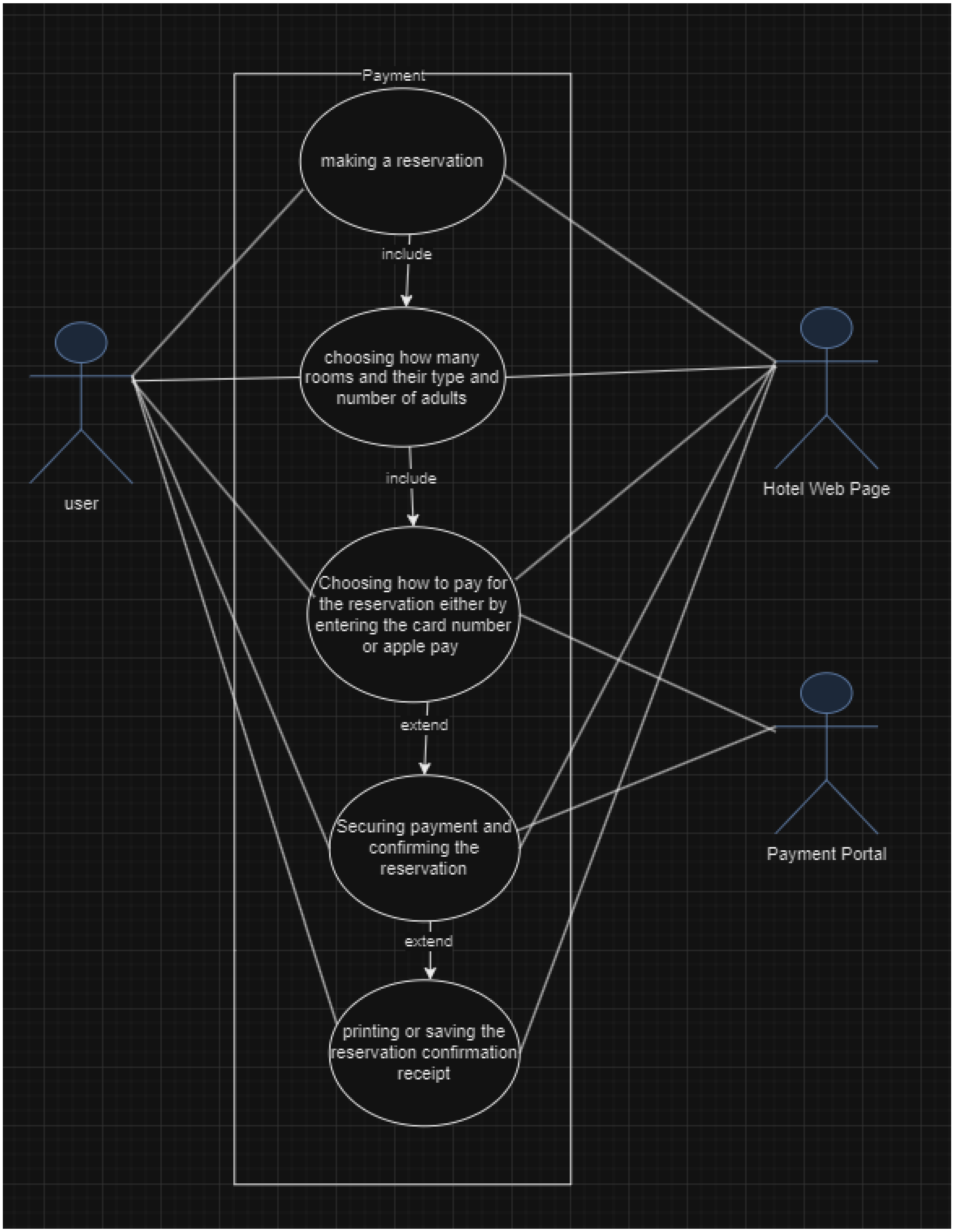
Part 1:

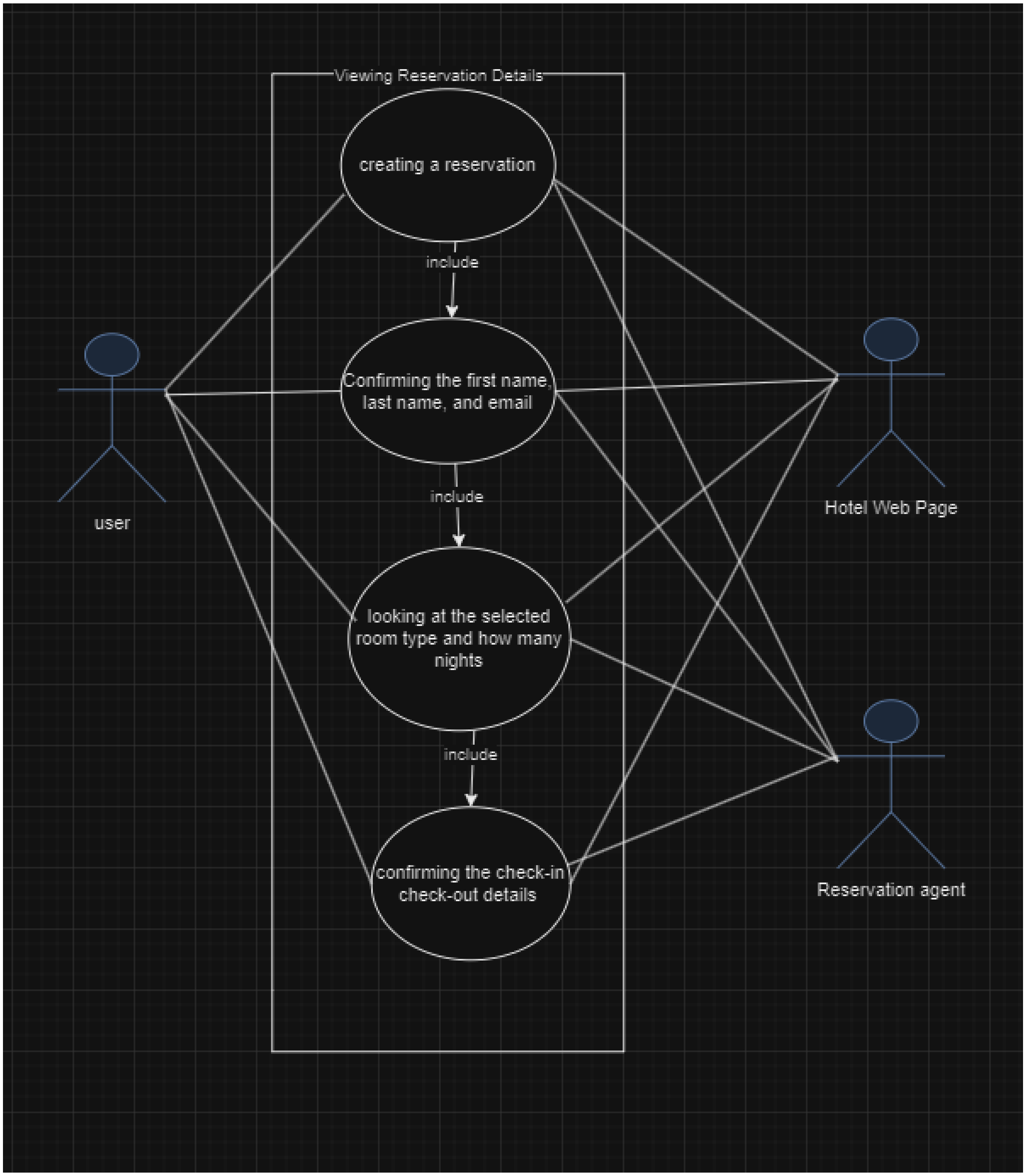
Software use cases:

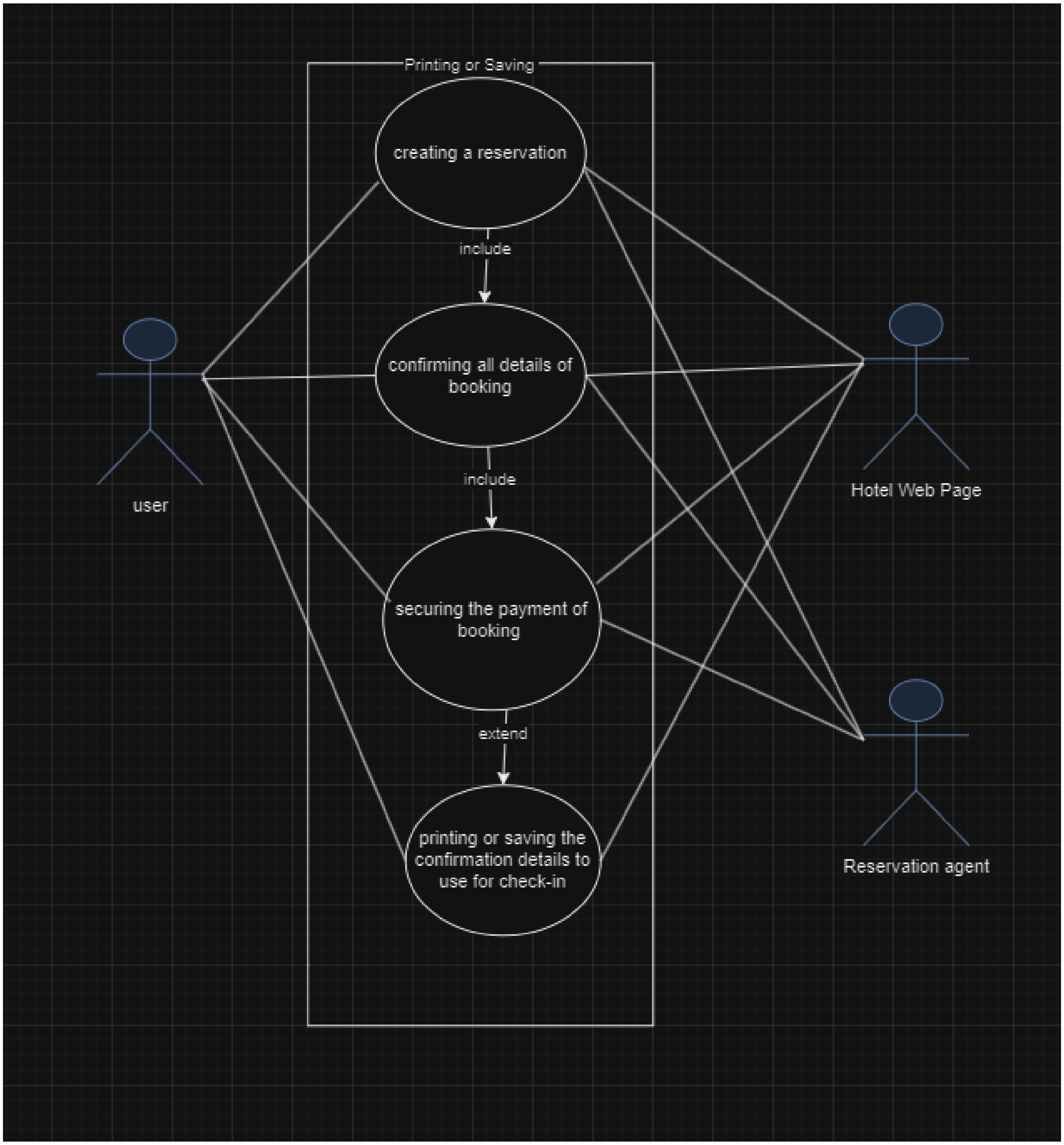
1. Create reservation: where the user can make a reservation by inputting multiple details like his name, the date, room type, and how many rooms.
2. View reservation details: where the user can see the confirmation of his booking where it displays the details such as the room type, check-in and check-out dates, number of nights, and the total charges.
3. Payment: The user inputs his credit card details so he could process the payment which includes the room subtotal, tax charge, and fees.
4. Printing or saving: The user has the option to get his reservation information sent to his email to be saved or getting it printed directly from the page to use for the check-in.

UML use-case diagrams:




Payment
making a reservation
include
choosing how many rooms and their type and number of adults
include
Choosing how to pay for the reservation either by entering the card number or apple pay
extend
Securing payment and confirming the reservation
extend
printing or saving the reservation confirmation receipt
user
Hotel Web Page
Payment Portal


Viewing Reservation Details
creating a reservation
include
Confirming the first name, last name, and email
include
looking at the selected room type and how many nights
include
confirming the check-in check-out details
user
Hotel Web Page
Reservation agent


Printing or Saving
creating a reservation
include
confirming all details of booking
include
securing the payment of booking
extend
printing or saving the confirmation details to use for check-in
user
Hotel Web Page
Reservation agent

Description Tables:

Use Case	Create reservation
Trigger	The user wants to book a room in a hotel
Preconditions	User has selected the hotel he wants and chose the available room type.
Main Scenarios	- User selects the hotel he wants and looks for the rooms - User chooses the check-in and check-out dates for his booking - User adds his personal details - The system verifies if there are any available selected rooms and checks the user's details. The hotel system confirms the reservation
Exceptions	1- The room selected is unavailable and the system shows other room types or dates. 2- The user’s details are not complete and the user has to fill in everything that is needed.

Use case	View reservation details
Trigger	The user wants to confirm his booking details
Preconditions	The booking is confirmed and does exist
Main scenarios	- User visits the home page and accesses his reservation - The system shows the reservation details that the user inputted - The user double-checks all the details before arrival
Exceptions	1- The user does not find his reservation and the system asks for details or a confirmation number to recheck. 2- The user selected the wrong check-in and check-out dates that need to be changed by contacting the hotel directly.

Use case	Payment
Trigger	The user has finished inputting details, now he needs to make the payment.
Preconditions	The user inserted valid booking details.
Main scenarios	- The user adds the card credentials to proceed with the payment - The system checks the card’s information - System confirms the payment and shows the digital receipt
Exceptions	1- The system rejects the card where there is not enough money in the card for the payment. 2- The payment portal has delays and asks to add card information again.

Use case	Printing or saving
Trigger	The user wants to print or save his digital receipt
Preconditions	The reservation was paid and confirmed
Main scenarios	- The receipt shows up and there is an option to save it on the device or to print it. - The user selected to print it first to keep a copy incase. - The user saves the digital one on his device to make it easier for him at check-in.
Exceptions	1- There is a technical issue that stopped the receipt from appearing and asks the user to wait. 2- The printer has no ink which means that he cant print the receipt.

Part 2:

Class: Reservation

Object: details

Description: This is all the details that the user entered for making a booking in a hotel.

Class: user

Object: type of user

Description: the person who wants to book a hotel room using a system.

Class: Hotel

Object: amenities

Description: the hotel where the reservation is made, having the location, name, and rating.

Class: Room

Object: type

Description: the type of room that was booked with its price, size, and if it is available.

Class: Payment

Object: card

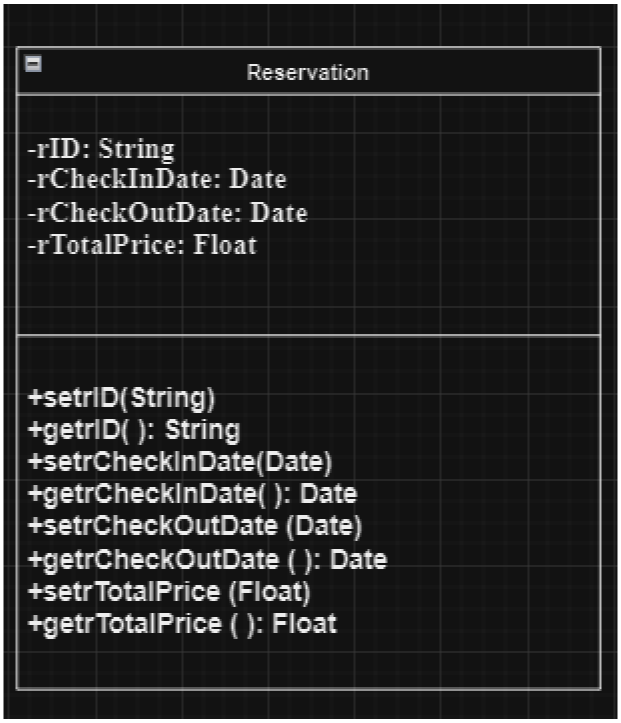
Description: this is the payment process where the user inputs his card credentials to pay for the reservation.

Class: Receipt

Object: information

Description: all the details of the reservation that was made goes in this receipt which includes the check-in and check-out dates, payment done, and the confirmation number.

UML Class Diagram:



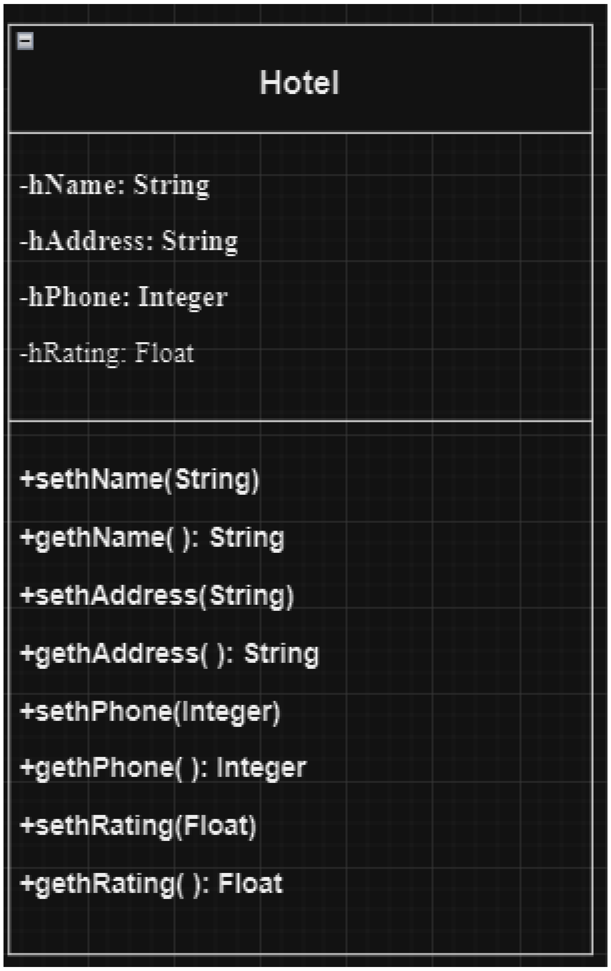
Reservation

-rID: String
-rCheckInDate: Date
-rCheckOutDate: Date
-rTotalPrice: Float

+setrID(String)
+getrID( ): String
+setrCheckInDate(Date)
+getrCheckInDate( ): Date
+setrCheckOutDate (Date)
+getrCheckOutDate ( ): Date
+setrTotalPrice (Float)
+getrTotalPrice ( ): Float

Description:

This diagram represents the reservation class where the attributes shown are the necessary details that the user needs to add to create a booking.



Hotel

-hName: String

-hAddress: String

-hPhone: Integer

-hRating: Float

+sethName(String)

+gethName( ): String

+sethAddress(String)

+gethAddress( ): String

+sethPhone(Integer)

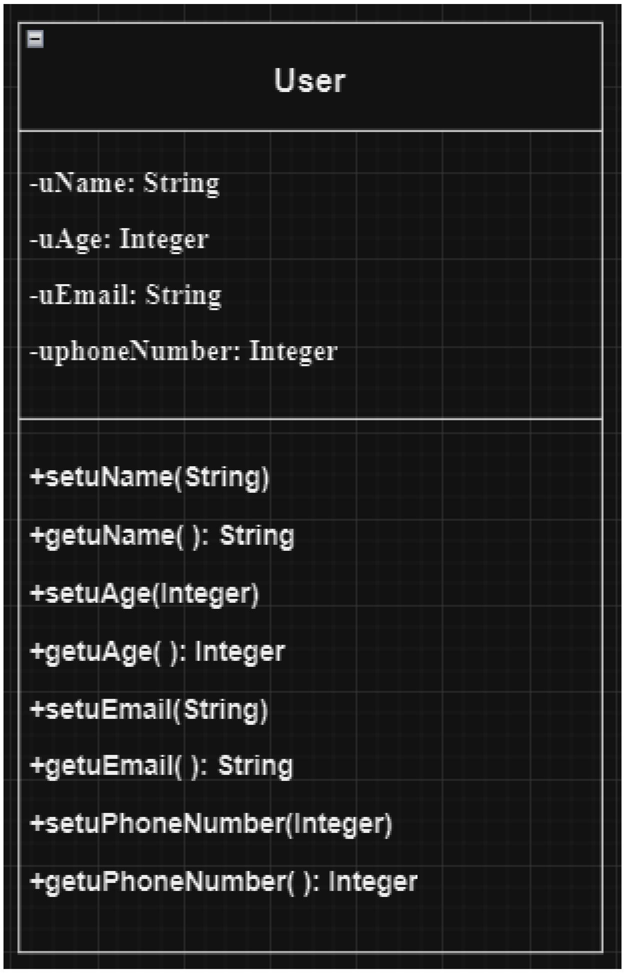
+gethPhone( ): Integer

+sethRating(Float)

+gethRating( ): Float

Description:

This diagram represents the hotel class where all the attributes shown are the specifications of the hotel where the user can view them to choose a suitable hotel.



User

-uName: String

-uAge: Integer

-uEmail: String

-uphoneNumber: Integer

+setuName(String)

+getuName( ): String

+setuAge(Integer)

+getuAge( ): Integer

+setuEmail(String)

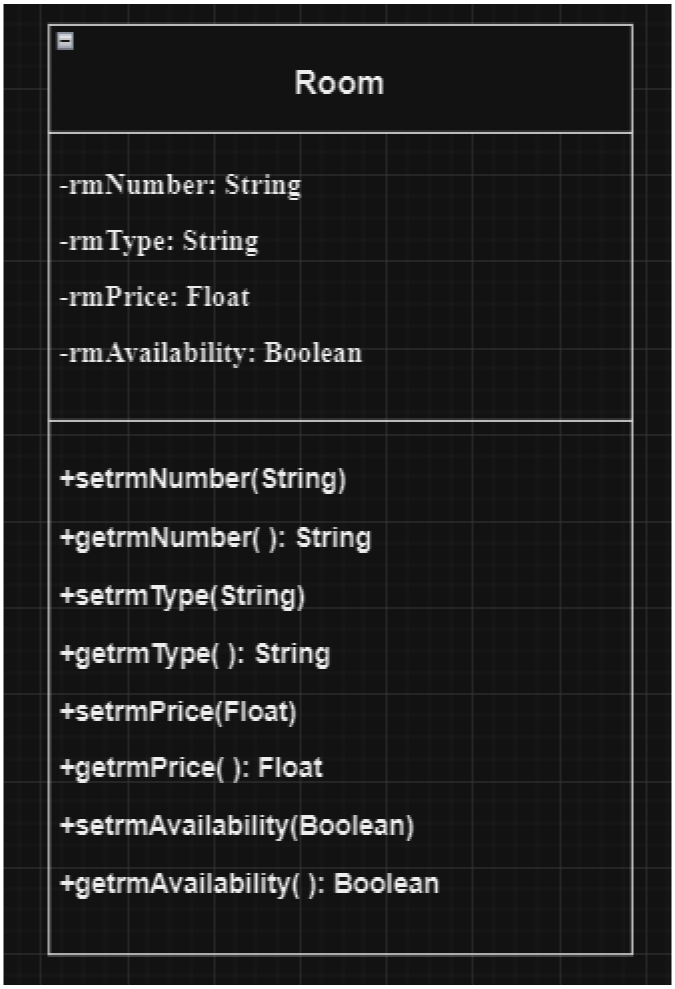
+getuEmail( ): String

+setuPhoneNumber(Integer)

+getuPhoneNumber( ): Integer

Description:

This diagram represents the person who is making a reservation using the hotel's system which is the user, where he has multiple personal information that is needed to be added in the reservation list to proceed.



Room

-rmNumber: String

-rmType: String

-rmPrice: Float

-rmAvailability: Boolean

+setrmNumber(String)

+getrmNumber( ): String

+setrmType(String)

+getrmType( ): String

+setrmPrice(Float)

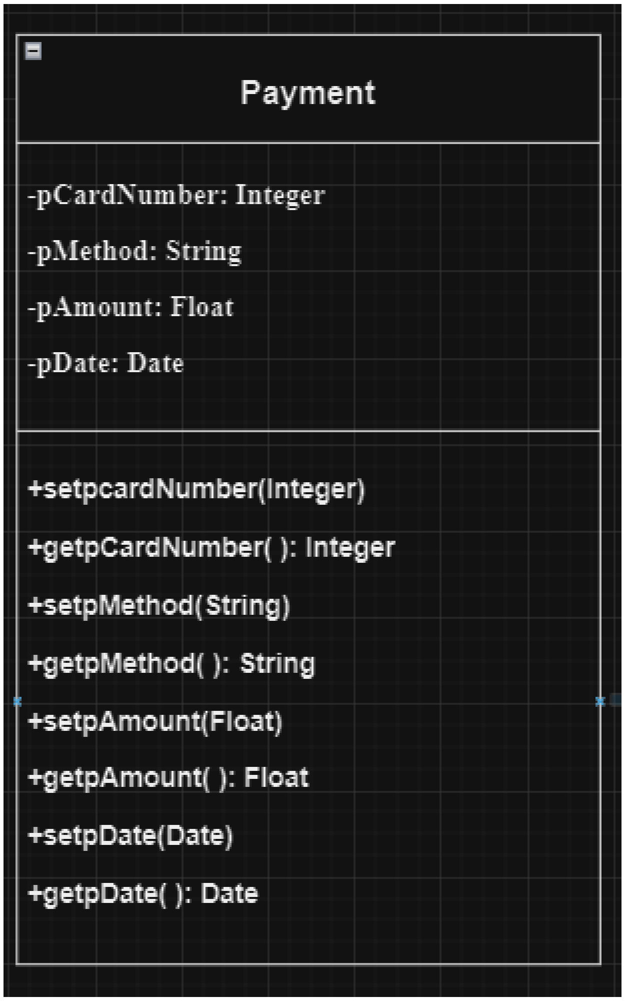
+getrmPrice( ): Float

+setrmAvailability(Boolean)

+getrmAvailability( ): Boolean

Description:

This diagram represents the different rooms in the hotel that are available, every room has its type, price per night, and to see if it is available or not depending on the user's preferences.



Payment

-pCardNumber: Integer

-pMethod: String

-pAmount: Float

-pDate: Date

+setpcardNumber(Integer)

+getpCardNumber( ): Integer

+setpMethod(String)

+getpMethod( ): String

+setpAmount(Float)

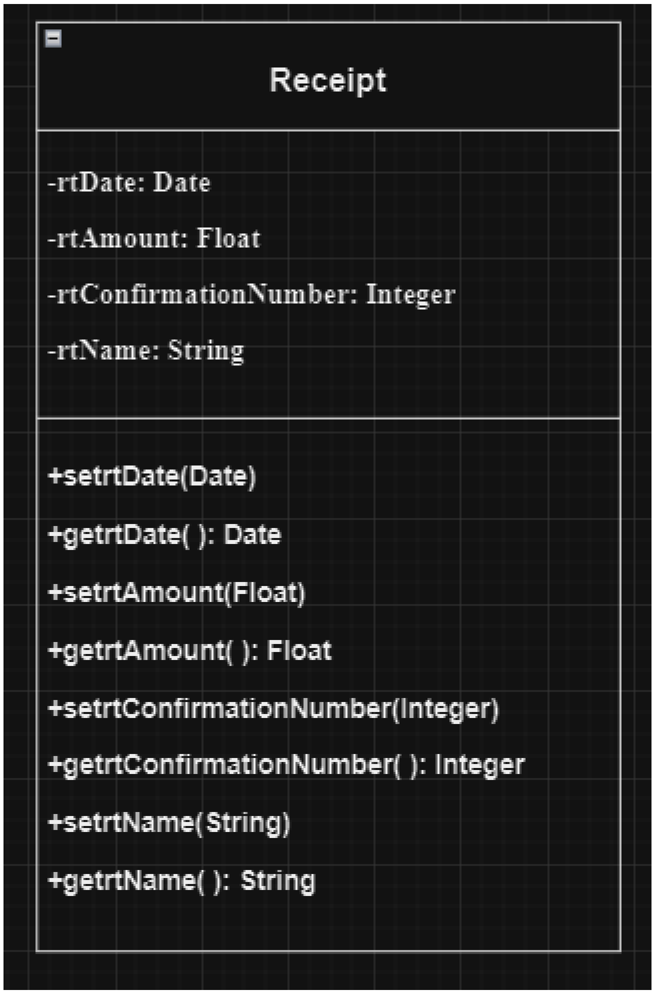
+getpAmount( ): Float

+setpDate(Date)

+getpDate( ): Date

Description:

This diagram represents the payment that is needed to make a reservation, for the user to proceed with everything and to get his confirmation number, he needs to pay or to put a card in the reservation details.



Receipt

-rtDate: Date

-rtAmount: Float

-rtConfirmationNumber: Integer

-rtName: String

+setrtDate(Date)

+getrtDate( ): Date

+setrtAmount(Float)

+getrtAmount( ): Float

+setrtConfirmationNumber(Integer)

+getrtConfirmationNumber( ): Integer

+setrtName(String)

+getrtName( ): String

Description:

This diagram represents the receipt class where for the user to have proof that he has a reservation done by him, he needs a receipt to use for check-in that has all the needed details.

Task 3: ALL THE CODES WERE INPUTED IN PYCHARM AND PASTED HERE

Reservation codes:

```
class Reservation:
    """This class represents the reservation of the user"""

    def __init__(self, reservationID, checkInDate, checkOutDate, totalPrice, room):
        self._reservationID = reservationID
        self._checkInDate = checkInDate
        self._checkOutDate = checkOutDate
        self._totalPrice = totalPrice
        self._room = room

    # Setters and Getters for the reservation attributes
    def set_reservationID(self, reservationID):
        self._reservationID = reservationID

    def get_reservationID(self):
        return self._reservationID

    def set_checkInDate(self, checkInDate):
        self._checkInDate = checkInDate

    def get_checkInDate(self):
        return self._checkInDate
```

```
def set_checkOutDate(self, checkOutDate):
    self._checkOutDate = checkOutDate

def get_checkOutDate(self):
    return self._checkOutDate

def set_totalPrice(self, totalPrice):
    self._totalPrice = totalPrice

def get_totalPrice(self):
    return self._totalPrice

def set_room(self, room):
    self._room = room

def get_room(self):
    return self._room

# Other required function-headers
def cancel_reservation(self):
    """This function will cancel the reservation."""
    pass

def modify_reservation(self, new_checkInDate, new_checkOutDate):
    """This function will modify the reservation check-in and check-out dates."""
    pass
```

Hotel codes:

```
class Hotel:
    """This class represents the Hotel that the user chose to stay in"""

    def __init__(self, hotelName, address, phone, rating, roomsAvailable):
        self._hotelName = hotelName
        self._address = address
        self._phone = phone
        self._rating = rating
        self._roomsAvailable = roomsAvailable

    # Setters and Getters for the attributeses
    def set_hotelName(self, hotelName):
        self._hotelName = hotelName

    def get_hotelName(self):
        return self._hotelName

    def set_address(self, address):
        self._address = address

    def get_address(self):
        return self._address
```

```
def set_phone(self, phone):
    self._phone = phone

def get_phone(self):
    return self._phone

def set_rating(self, rating):
    self._rating = rating

def get_rating(self):
    return self._rating

def set_roomsAvailable(self, roomsAvailable):
    self._roomsAvailable = roomsAvailable

def get_roomsAvailable(self):
    return self._roomsAvailable

# Other required function-headers
def check_availability(self):
    """This function should check if the rooms are available."""
    pass

def get_hotel_info(self):
    """This function should return a summary of the hotel's information like reviews."""
    pass
```

User codes:

```
class User:
    """This class represents the user who is trying to make a reservation"""

    def __init__(self, userID, firstName, lastName, email, phoneNumber):
        self._userID = userID
        self._firstName = firstName
        self._lastName = lastName
        self._email = email
        self._phoneNumber = phoneNumber

    # Setters and Getters for the attributes
    def set_userID(self, userID):
        self._userID = userID

    def get_userID(self):
        return self._userID

    def set_firstName(self, firstName):
        self._firstName = firstName

    def get_firstName(self):
        return self._firstName

    def set_lastName(self, lastName):
```

```
    self._lastName = lastName

def get_lastName(self):
    return self._lastName

def set_email(self, email):
    self._email = email

def get_email(self):
    return self._email

def set_phoneNumber(self, phoneNumber):
    self._phoneNumber = phoneNumber

def get_phoneNumber(self):
    return self._phoneNumber

# Extra Function-headers
def update_profile(self):
    """This function should manage updating the user's profile details."""
    pass

def delete_account(self):
    """This function should handle the process of erasing the user's account from the system."""
    pass
```

Room codes:

```
class Room:
    """This class represents the room type that the user specified"""

    def __init__(self, roomNumber, roomType, pricePerNight, isAvailable, bedType):
        self._roomNumber = roomNumber
        self._roomType = roomType
        self._pricePerNight = pricePerNight
        self._isAvailable = isAvailable
        self._bedType = bedType

    # Setters and Getters for the attributes
    def set_roomNumber(self, roomNumber):
        self._roomNumber = roomNumber

    def get_roomNumber(self):
        return self._roomNumber

    def set_roomType(self, roomType):
        self._roomType = roomType

    def get_roomType(self):
        return self._roomType

    def set_pricePerNight(self, pricePerNight):
        self._pricePerNight = pricePerNight

    def get_pricePerNight(self):
        return self._pricePerNight

    def set_isAvailable(self, isAvailable):
        self._isAvailable = isAvailable

    def get_isAvailable(self):
        return self._isAvailable

    def set_bedType(self, bedType):
        self._bedType = bedType

    def get_bedType(self):
        return self._bedType

    # extra required function-headers
    def book_room(self):
        """This function should book the room if it is available."""
        pass

    def cancel_booking(self):
        """This function should cancel the room booking where it is not available."""
        pass
```

Payment codes:

```
class Payment:
    """This class represents the payment of the user for the hotel to make his reservation"""

    def __init__(self, paymentID, paymentMethod, amount, paymentDate, cardNumber):
        self._paymentID = paymentID
        self._paymentMethod = paymentMethod
        self._amount = amount
        self._paymentDate = paymentDate
        self._cardNumber = cardNumber

    # Setters and Getters for the attributes above
    def set_paymentID(self, paymentID):
        self._paymentID = paymentID

    def get_paymentID(self):
        return self._paymentID

    def set_paymentMethod(self, paymentMethod):
        self._paymentMethod = paymentMethod

    def get_paymentMethod(self):
        return self._paymentMethod

    def set_amount(self, amount):
        self._amount = amount

    def get_amount(self):
        return self._amount

    def set_paymentDate(self, paymentDate):
        self._paymentDate = paymentDate

    def get_paymentDate(self):
        return self._paymentDate

    def set_cardNumber(self, cardNumber):
        self._cardNumber = cardNumber

    def get_cardNumber(self):
        return self._cardNumber

    # Other required function-headers
    def process_payment(self):
        """This function should secure and process the payment."""
        pass

    def refund_payment(self):
        """This function should handle refunding the payment to the user."""
        pass
```

Receipt codes:

```
class Receipt:
    """This class represents the receipt of the user"""

    def __init__(self, receiptID, reservation, totalAmount, dateIssued, paymentMethod):
        self._receiptID = receiptID
        self._reservation = reservation
        self._totalAmount = totalAmount
        self._dateIssued = dateIssued
        self._paymentMethod = paymentMethod

    # Setters and Getters for the attributes
    def set_receiptID(self, receiptID):
        self._receiptID = receiptID

    def get_receiptID(self):
        return self._receiptID

    def set_reservation(self, reservation):
        self._reservation = reservation

    def get_reservation(self):
        return self._reservation

    def set_totalAmount(self, totalAmount):
```

```
    self._totalAmount = totalAmount

def get_totalAmount(self):
    return self._totalAmount

def set_dateIssued(self, dateIssued):
    self._dateIssued = dateIssued

def get_dateIssued(self):
    return self._dateIssued

def set_paymentMethod(self, paymentMethod):
    self._paymentMethod = paymentMethod

def get_paymentMethod(self):
    return self._paymentMethod

# Other required function-headers
def generate_receipt(self):
    """This function should show the receipt details."""
    pass

def print_receipt(self):
    """This function should handle printing the receipt if the user needed it."""
    pass
```

Task 4:

Creating the objects:

```
# Creating objects for each of the classes I did above

# creating User object
user = User(1, "Ted", "Vera", "tedvera@mac.com", "505-661-1110")

# creating Hotel object
hotel = Hotel("Comfort Inn & Suites Los Alamos", "2455 Trinity Drive, Los Alamos, NM 87544", "505-661-1110", 4.5, 20)

# creating Room object
room = Room("105", "2 Queen Beds", 89.95, True, "Queen")

# creating Reservation object
reservation = Reservation("52523887", "Sun, Aug 22, 2010 - 03:00 PM", "Tue, Aug 24, 2010 - 12:00 PM", 201.48, 1)

# creating Payment object
payment = Payment("111342", "Mastercard (ending in 9904)", 201.48, "22-08-2010", "9904")

# Displaying the information given in figure 1

print("Your Reservation Is Confirmed")
print(f"Thank you for your reservation. Please print your hotel receipt and show it at check-in.")
print(f"Your Name: {user.get_firstName()} {user.get_lastName()}")
print(f"Your Email: {user.get_email()}")
print(f"Hotel Confirmation Number: {reservation._reservationID}\n")

print(f"{hotel.get_hotelName()}")
print(f"{hotel.get_address()}")
print(f"Check-In: {reservation.get_checkInDate()}")
print(f"Check-Out: {reservation.get_checkOutDate()}")
print(f"Room: {room._roomNumber}, {room.get_roomType()}")
print(f"Room Type: {room.get_roomType()}\n")

print("Summary of Charges")
print(f"Billing Name: {user.get_firstName()} {user.get_lastName()}")
print(f"Credit Card: {payment.get_paymentMethod()}")
print(f"Room Cost (per night): ${room.get_pricePerNight()}")
print(f"Room Subtotal (2 nights): ${room.get_pricePerNight() * 2:.2f}")
print(f"Taxes and Fees: ${payment.get_amount() - (room.get_pricePerNight() * 2):.2f}")
print(f"Total Charges: ${payment.get_amount():.2f}")
```

GITHUB LINK:

Summary:

Through this assignment, I learned how to create different types of diagrams such as the UML case-diagram, UML class diagram, and the UML case-diagram description tables. I learned how to use different tools through the diagram app that was provided to us by our instructor. I learned how to create python codes using the pycharm app which helped me a lot by moving step-by-step to finish all the requirements needed to create all the codes for all the classes and attributes and objects. I would like to learn how to use different tools to create new diagrams in the future and would like to learn how to generate codes that are specified for these types of diagrams.